Proposals for Enhancing the FITS Data Format

William Pence, Lucio Chiappetti, Malcolm Currie, Jessica Mink, Rob Seaman

Version 0.6~~4~~

26~~29~~ March~~January~~ 2014

0 Overview

The FITS scientific data format was invented in 1979 and has greatly benefited astronomical research by providing a common data interchange format that is used by virtually all scientists and institutions in the field. FITS continues to serve astronomers well and will likely still be used for decades to come. It is widely acknowledged, however, that the FITS format contains various limitations that in part reflect the state of computer systems at the time FITS was created (e.g., the prevalence of sequential magnetic tape for data storage and the scarcity of available computer memory). In this paper, we discuss ways in which the FITS format could be enhanced to eliminate some of these limitations. We specifically restrict this discussion to relatively minor changes to the FITS format that would not cause much disruption to existing software systems and would be relatively easy to implement.

In the following sections of this paper, we discuss several specific issues regarding the FITS format:

1. The limit on the length of keyword names
2. The limit on the length of character string keyword values
3. The restriction on the set of characters that may be used in a keyword name

For each of these issues, we, a) describe the impact that the current restriction has on FITS users, b) propose simple changes to the definition of the FITS format to remove that restriction, and c) analyze the cost to the FITS community of implementing that change.

Our main goal in producing this document is to stimulate further discussion within the FITS community of ways in which the FITS format can be improved for the benefit of current and future users. We hope that these discussions will ultimately lead to specific proposals for changes to the FITS Standard document that can be submitted to the IAU FITS Working Group for further consideration.

1 The 8-character limit on the length of header keyword names

Because FITS header records have an 80-byte fixed format that requires the equals sign character, which separates the keyword name from the keyword value, to occur in byte 9, this limits keyword names to a maximum length of 8 characters. In this section we discuss the impact that this restriction has on FITS users and propose a simple change to the FITS format that will allow longer keyword names. We also discuss the impact and cost to the FITS community of implementing this change.

1.1 Effects of the 8-character keyword name limit

While limiting keyword names to 8 characters is sometimes convenient (e.g., when allocating space to store or display a list of keyword names) this restriction[1] negatively affects FITS users in several significant ways:

1.1.1 Increases the potential for confusion

Limiting keyword names to 8 characters forces the designers of FITS files to invent cryptic mnemonic names that are confusing to both software developers and to end users of the data files. Examples of such keyword names can be found in the FITS data files produced by virtually every astronomical project or observatory. To cite just a couple real examples, instead of the cryptic `UCH2CJTM` keyword, one could instead have a much more descriptive name such as `TEC_COLD_JUNCTION_2_TEMP`, and instead of `ROTRTTRG`, one could have a keyword called `TARGET_ROTATION_RATE`.

This length restriction is even more severe when dealing with arrays of indexed keywords such as `VOLTAG1`, `VOLTAG2`, ..., because the presence of the sequence number leaves fewer characters available for the root name of the keyword. Similarly, the root name of a keyword associated with a specific column in a FITS table (e.g., `TUNITn`) is limited to at most 5 characters to allow room for the 3-digit column number suffix. The World Coordinate System (WCS) keywords (as defined in Table 22 of version 3 of the FITS Standard) serve as a good illustration of this problem. In several instances, the root name of the WCS keyword is limited to just 2 characters (e.g., `12PC104A`) and in the most extreme case, the root name is reduced to a single letter (e.g., the letter '`V`' in `7V104_9A`) because of the need to also encode the column number, up to 2 coordinate axis numbers, a parameter sequence number, and the alternate WCS version code letter, all with only 8 characters! If and when longer keyword names are supported in FITS, one immediate beneficial use of them would be to create more descriptive aliases for these cryptic 8-character WCS keyword names.

1.1.2 Hinders the development of new innovative conventions

The keyword length limitation hinders innovation in developing new conventions for representing the more complex types of data products that are being generated by current and future astronomical instruments. For example, FITS binary tables offer great flexibility in storing complex data structures, however, the limitations on keyword names are an obstacle to representing the meta-data that is needed to fully describe those data structures. For example, the current 8-character limit severely restricts the ability to construct hierarchical keyword naming conventions or to establish a 'namespace' for a designated set of keywords by adding an easily identifiable prefix to the keyword names. To cite another WCS example, one reason that the draft paper on representations of distortions in WCSs has take so long to be completed is because it became necessary to invent an entirely new type of `record-valued' keyword structure because it is impossible to represent all the possible distortion keywords in any sensible way with 8-characters names. This problem could be eliminated simply by allowing longer keyword names.

---

1 While the 8-character limit is restrictive by today's standards, it was actually a significant increase over the 6-character limit on variable names in the Fortran-77 computer language which formed the basis of much of the FITS header record syntax.

1.1.3 Hinders the exchange of data from different formats

The keyword name length restriction is also an obstacle when attempting to convert data files from other scientific formats (e.g., HDF5 or CDF) into FITS format. Since most other data formats allow meta-data parameters with symbolic names longer than 8 characters, representing those parameters as FITS header keywords is problematic. This interferes with one of the primary purposes of the FITS format which is to serve as a interchange format for transporting data from one data analysis environment to another.

1.2 Proposed convention for supporting longer keyword names

The most obvious and direct way to support longer FITS keyword names is to relax the fixed-format syntax of the 80-character keyword records so that the equals sign followed by a space character (the so-called 'value indicator' that separates the keyword name from the value field) is not required to be in bytes 9-10 of the keyword record and instead may be located anywhere in or beyond that position. While this could in principle allow names up to 77 characters long (leaving only a single character for the value field), we suggest that the keyword name be limited to 55~~6~~ characters, which then leaves at least 23~~2~~ characters for the value field. This is enough space to represent the longest 64-bit integer values (20 characters) as well as floating point numbers in exponential notation with 16 decimal places of precision (23~~2~~ characters) as shown here:

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
KEYWORD_AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA~~A~~= -1.234567890123456E-123
```

Note that arbitrarily long character string keyword values can also be assigned to keywords with long names by using a continuation convention, as is discussed in Section 2 of this document.

It will be assumed here that the longer keyword names will inherit the same restrictions that apply to the existing 8-character FITS keywords and thus may contain only the uppercase letters A-Z, the digits 0 - 9, and the minus sign and the underscore character, with no embedded spaces. In Section 3 of this paper, we discuss a proposal for expanding the allowed set of characters in keyword names.

Finally, we note in passing that the existing ESO HIERARCH keyword convention (see http://fits.gsfc.nasa.gov/registry/hierarch_keyword.html) could in principle be used to represent keywords with long names, with a syntax such as,

```
HIERARCH LONG_KEYWORD_NAME = 1.234 / comment field
```

The CFITSIO library, in fact, uses this convention if an application program attempts to write a non-standard keyword with a name longer than 8 characters. Regardless, we do not recommend that the HIERARCH convention be used more generally to represent long keyword names for several reasons: it abuses the original intent of this convention which is to represent sets of keywords that have a hierarchical name structure; it would likely be confusing to users to reuse this convention for an entirely different purpose; and it is an inefficient way of representing long keyword names because the '`HIERARCH `' string uses up 9 characters in the keyword record that could otherwise be used in the keyword name itself.

1.3 Impact and costs of supporting longer keyword names

It is important to note that the FITS Standard already allows keywords that have the new format that is being proposed here[2]. The only change being proposed is in the way that these keywords should be interpreted. Under the current rules, these keyword records should simply be interpreted as containing commentary text, similar to `COMMENT` or `HISTORY` keywords, because the `'= '` value indicator is not located in bytes 9 and 10 of the record and thus there is no value field. Under this new proposal, instead, the value indicator is allowed to be located anywhere between bytes 9-10 and bytes 56~~7~~-57~~8~~, inclusive. and the bytes preceding it are interpreted as containing the keyword name, and the bytes following it are interpreted as containing the value field (and an optional comment field).

Since these 'free-format' style keyword records have always been allowed, one would expect that their presence in FITS headers should have no adverse affect on existing FITS software. To verify this, we tested a variety of existing software packages, including FTOOLS, DS9, WCStools, [add other examples here], using a FITS image file that contains several keywords with long names. We found no cases where these keywords caused the software to behave abnormally.

The main effect of this proposed change to FITS will be that legacy software applications will not recognize the long keyword names and will be unable to correctly interpret the corresponding keyword value until such time as the software is modified to support this new keyword format. This impact will be mitigated to a certain extent, however, by the likely fact that these new long keyword names will initially mainly be used by new projects for mission-specific keywords. These keywords will generally only need to be read or written by the analysis software for that particular mission. Existing general purpose FITS utility programs (e.g., Sextractor, or DAOPHOT) will most likely not need to read or write keywords with long names in order to perform their function. Thus, most existing software programs can safely ignore any keywords with long names.

We estimate that the cost of retrofitting existing software to fully support longer keyword names to be relatively small. The biggest impact will be on the few dozen or so FITS libraries that most applications software packages rely on to read and write FITS files (see http://fits.gsfc.nasa.gov/fits_libraries.html). While some libraries will obviously require more effort than others to be upgraded, in three cases that we have examined in detail (the CFITSIO library, the WCStools FITS library, and the TAM Java FITS library), the developers of those libraries have estimated that the modifications needed to support longer keyword names could be implemented with only a couple days of programmer effort.

Once a FITS library has been upgraded, the applications programs that are linked to that library will then largely inherit the ability to read and write keywords that have longer names. In some cases, however, it may also be necessary to make further small modifications to the application program itself, for example, to increase the size of string variables that are used to store the longer keyword names (e.g., change CHARACTER*8 Fortran declarations to CHARACTER*55~~6~~). These modifications are not complex in nature, so many projects should be able to absorb this level of effort as part of the normal software maintenance activities.

1.4 Summary

The 8-character keyword name limitation has a significant negative impact on current FITS users by causing confusion and impeding the development of new FITS conventions. Although existing projects

[2] Section 4.1.2.3 states: "In keyword records without a value indicator [in bytes 9 - 10], bytes 9 through 80 should be interpreted as commentary text, however, this does not preclude conventions that interpret the content of these bytes in other ways."

have successfully coped with the resulting cryptic 8-character keyword names, removing this limitation will provide significant benefits to future projects by allowing them to create clearer, more self-documenting FITS files. Given that the cost of upgrading software to support longer keyword names is relatively small, this appears to be a very worthwhile investment for enhancing future astronomical research.

2 The 68-character limit on string-valued header keywords

The second FITS format restriction to be discussed here is the 68-character limit on the length of string-valued keywords (where in this maximum length case, the delimiting quote characters of the string are in bytes 11 and 80 of the header record). While this length is sufficient for many uses, almost every producer of FITS data files has encountered cases where it would desirable to represent character strings that are longer than this arbitrary 68-character limit. When faced with this dilemma, FITS file designers have usually resorted to various work-arounds as described in the following sections.

2.1 Work-around 1: Invent a local convention

Since there is no standard FITS convention for representing long text strings in FITS headers, some projects that need this capability have invented their own local convention for this purpose. Two known local FITS conventions (there may be others as well) are described below:

2.1.1 The IRAF local convention for WCS distortion coefficients

The IRAF project invented a keyword convention to represent the polynomial coefficients that are needed to compute celestial coordinates when using the TNX world coordinate system that looks like this:

```
WAT1_001= 'wtype=tnx axtype=ra lngcor = "3. 4. 4. 2. -0.3171856965643079 -0.015'
WAT1_002= '0652479325533 -0.3126038394350166 -0.1511955040928311 0.002318100364'
WAT1_003= '838772 0.01749134520424022 -0.01082784423020123 -0.1387962673564234 '
WAT1_004= '-4.307309762939804E-4 0.009069288008295441 0.002875265278754504 -0.0'
WAT1_005= '4487658756007625 -0.1058043162287004 -0.06862147653757 67 "         '
```

Software that needs to calculate TNX coordinate transformations must be programmed to read and concatenate the values of all the `WAT1_n` keywords into a long string that is then parsed to determine the values for the individual coefficients. While this convention is adequate for the specific purpose of recording WCS distortion coefficients, it is not easily extensible for more general use. The main problem is that the syntax of this convention provides no indication to the reading software that the array of keywords actually represent one long string value instead of just representing a set of normal independent string-valued keywords.

2.1.2 The HEASARC long-string local convention

In the 1990s, the HEASARC developed a more general convention for representing any character-string keyword value that is too long to represent within a single header record. Under this HEASARC convention, the ampersand character (`&') is appended to each sub-string as a flag to indicate that it is continued on the next header record, and each continued record has the keyword name `CONTINUE`, as shown by the following examples taken from some high-energy astrophysics FITS files:

Example 1: the internal configuration of the complex Rossi X-Ray Timing Explorer instrument is described with keywords such as this:

```
TDDES12 = 'D[0~3] & E[0~63] & T[0;1;16] & C[(S[msLimit1]~S[msLimit2]),((S[msLi&'
CONTINUE  'mit2]+1)~S[msLimit3]),((S[msLimit3]+1)~S[msLimit4]),((S[msLimit4]+1&'
CONTINUE  ')~S[msLimit5])] '
1CPIX12 = '(S[msLimit1]~S[msLimit2]),((S[msLimit2]+1)~S[msLimit3]),((S[msLimit&'
CONTINUE  '3]+1)~S[msLimit4]),((S[msLimit4]+1)~S[msLimit5])'
```

Example 2: The Chandra X-ray mission uses the CONTINUE header keyword convention to store the full directory path to various calibration files:

```
BPIXFILE= '/dsops/ap/sdp/opus/prs_run/tmp//ACIS_F_L1_08784n137/output/acis&'
CONTINUE  'f00317_000N001_bpix1.fits'
BIASFIL0= '/dsops/ap/sdp/opus/prs_run/tmp//ACIS_F_L1_08784n137/input/acisf&'
CONTINUE  '087679048N001_0_bias0.fits'
BIASFIL2= '/dsops/ap/sdp/opus/prs_run/tmp//ACIS_F_L1_08784n137/input/acisf&'
CONTINUE  '087679048N001_5_bias0.fits'
BIASFIL3= '/dsops/ap/sdp/opus/prs_run/tmp//ACIS_F_L1_08784n137/input/acisf&'
CONTINUE  '087679048N001_4_bias0.fits'
BIASFIL5= '/dsops/ap/sdp/opus/prs_run/tmp//ACIS_F_L1_08784n137/input/acisf&'
CONTINUE  '087679048N001_2_bias0.fits'
BIASFIL6= '/dsops/ap/sdp/opus/prs_run/tmp//ACIS_F_L1_08784n137/input/acisf&'
CONTINUE  '087679048N001_3_bias0.fits'
BIASFIL7= '/dsops/ap/sdp/opus/prs_run/tmp//ACIS_F_L1_08784n137/input/acisf&'
CONTINUE  '087679048N001_1_bias0.fits'
```

This HEASARC long-string convention has mainly been used within the high energy astrophysics community and is only supported by a few of widely used FITS I/O libraries.

2.2 Work-around 2: Use commentary keywords

Rather than inventing a new convention, the most frequently used work-around for recording long character strings in FITS headers is simply to write them as a series of `COMMENT` or `HISTORY` keywords. While this is a suitable solution when the text is intended mainly for human consumption, it is not ideal when software needs to access and interpret that information because these commentary keywords do not formally have a `value'; the entire keyword is considered to contain unstructured commentary information. Also, the specific keyword of interest can be difficult to locate when there may be a large number of other commentary keywords in the header.

To illustrate this more clearly, the following examples show how important textual information that is currently represented using the ubiquitous `HISTORY` or `COMMENT` keywords, could be more clearly and definitively expressed as the value of a uniquely named keyword:

Example 1: Many FITS images generated by the National Radio Astronomy Observatory contain a set of `HISTORY` keywords that stipulate the legal licensing requirements for using the data:

```
HISTORYNRAO/VLAArchiveSurveyimages(NVASforshort)come"asis",withno
HISTORYwarrantyandmayonlybeusedforscientificandpersonalpurposes.
HISTORYUsage,publicationandredistributionforscientificpurposesisfree
HISTORYofcharge,providedthattheproperimagecreditbelowisincluded.
HISTORYAnyotherusageofNVASimagesMUSTbeapprovedbytheNRAOdirector.
```

This very important legal information could be more prominently highlighted by expressing it as the value of a unique keyword, such as:

```
LICENSE = 'NRAO/VLA Archive Survey images (NVAS for short) come "as is" ...
```

where the remaining licensing text would then be continued using some officially recognized FITS convention. The NRAO documentation related to these files could then unambiguously refer users to the LICENSE keyword for this essential information, instead of users having to search for the information within the many other HISTORY keywords in the file. It would also be straightforward to program software to highlight these requirements to users of the data (and even force users to acknowledge that they have read and agree to these licensing requirements), if so desired.

Example 2: Many existing FITS files contain a pair of COMMENT keywords that provide a bibliographic reference to further information about the FITS data format:

```
COMMENT   FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT   and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
```

This information could be more clearly differentiated from all the other COMMENT keywords in the file and could be made more easily accessible by software by creating a unique keyword for this purpose, such as:

```
FITSINFO='FITS(FlexibleImageTransportSystem)formatisdefinedin...
```

where, again, the text string would be continued using a standard FITS convention for representing long string values.

While some might consider these trivial examples, with a little effort one can surely find other cases in existing FITS files where blocks of commentary keywords have been used to represent information that would have been more effectively presented as the value of a unique keyword. Other examples might include: a summary of the weather conditions during the observation; the observer's log-book entries at the time of the observation; a 'figure caption' that describes the objects within a FITS image; the full directory path to associated data files; or the full postal address of the observer or the observatory. This is not to suggest that COMMENT and HISTORY keywords should not be used when appropriate, however it seems unreasonable for FITS file creators to be limited to this approach simply because the character string happens to be more than 68 characters long. Thus, we believe that a standard FITS convention needs to be defined for representing long string keyword values when the need arises.

2.3 Proposed conventions for long string-valued FITS keywords

The simplest and most obvious way to record arbitrarily long text strings within systems, such as FITS headers, that have underlying fixed-length records is to define a convention for continuing the text over multiple header records. There are 2 necessary parts to such a convention: 1) define a unique 'continuation character' to be appended to the end of each sub-string to indicate that it is to be continued[3], and 2) define a convention for the name that is assigned to the continued keyword records (e.g., the HEASARC convention uses the '&' continuation character and uses CONTINUE as the continuation keyword name). Out of the vast number of possible continuation conventions that could be devised, we recommend that the FITS community either adopt the existing HEASARC convention 'as is', or choose a slight improved variation on that convention. These two choices are discussed in the following 2 sections.

3 For reference, Fortran 90 uses '&' as the continuation character; C macros, JavaScript, and Python all use '\'; visual basic uses '_'; Windows PowerShell uses the back-tick '`'; Octive uses 3 dots '...'; IDL uses '$'; MS-DOS batch files use '^'; Rexx uses a comma ','; FTP and Easytrieve use '+'; and Xspec macros use '-'.

2.3.1 The HEASARC CONTINUE convention

The HEASARC CONTINUE keyword convention has been in use within the high-energy astrophysics community since 1994. A detailed description of this convention can be found on the FITS Support Office Web site at http://fits.gsfc.nasa.gov/registry/continue_keyword.html. In brief, under this convention the long text string is first divided into a series of sub-strings that are each no more than 67 characters long. The '&' continuation character is appended to the end of each sub-string (except for the last one) then the whole string is enclosed within single quote characters. Each sub-string is then written, in order, to as many consecutive header keywords as needed. The first keyword in the sequence has the user specified name, and all the following continuation keywords have the name `CONTINUE`. One feature of this convention is that the `CONTINUE` keywords do not have an equals sign in byte 9 of the header record, so they do not formally contain a value field except as defined within this convention. This makes it less likely that other FITS software that does not understand this continuation convention will modify or otherwise corrupt the continued strings. Here is a simple example:

```
ABSTRACT= 'Fifteen spirals are now available for which the sense of the spiral&'
CONTINUE ' pattern and the sense of the spectrographic rotation are known an&'
CONTINUE 'd in which there is conspicuous dissymmetry of obscuration.'
```

The main advantage of adopting this convention is that it already has a proven track record with over 20 years of use with no significant problems. Also, the fact that this convention is already supported by several widely used FITS libraries (including CFITSIO and the TAM Java FITS library), will reduce the overall implementation cost to the FITS community as compared to adopting an entirely new continuation convention.

One potential weakness of the HEASARC continuation convention, however, is that it implicitly depends on the ordering of the header keyword records. If the keyword order is not preserved, then it may not be possible to reconstruct the correct long string values. It should be noted, however, that a new recommendation was added to Version 3 of the FITS Standard (in Section 4.1.1), in part to address this concern: "It is recommended that the order of the keywords in FITS files be preserved during data processing operations because the designers of the FITS file may have used conventions that attach particular significance to the order of certain keywords ...". While this issue still remains a theoretical concern, in practice the HEASARC is not aware of any actual cases where the long-string values have been corrupted due to the keyword order not being preserved.

2.3.2 A more robust continuation convention

The alternative continuation convention described in this section improves on the HEASARC convention by explicitly appending a sequence number to the name of each continued header record which makes it possible to reconstruct the long string values even if the order of the keywords in the FITS header is changed. The basic format of this convention is shown in this example:

```
ABSTRACT= 'Fifteen spirals are now available for which the sense of the spiral&'
ABSTRACT_1 ' pattern and the sense of the spectrographic rotation are known an&'
ABSTRACT_2 'd in which there is conspicuous dissymmetry of obscuration.'
```

The only difference from the HEASARC convention is that instead of using keywords named `CONTINUE` for each continuation header record, this convention appends an underscore character followed by a sequence number to the original keyword name. Since this may result in keyword names longer than 8 characters, this continuation convention also requires support for the long keyword name convention, as discussed in Section 1 of this paper.

We defer to the IAU FITS Working Group in the choosing which of these (or other) continuation convention to adopt, but our slight preference would be for the HEASARC convention, simply because it is already used (and will necessarily continue to be used) in some segments of the FITS community, and thus it might be confusing to introduce a second continuation convention. [Should we vote on this??]

2.4 Impact and costs of supporting longer keyword string values

One impact of supporting long keyword value strings will be that programmers may need to take additional care in managing the computer memory needed to temporarily store the long string values. Under the current limitations of FITS keywords, programmers can rely on the fact that string keyword values will never require more than 68-bytes of storage space, but when the continuation convention is used, the strings may be arbitrarily long. Depending on the particular computer language that is used, the programmer may need to take additional steps in the code to either pre-allocate enough memory to store the string value, and/or free that memory when the string value is no longer needed. Whether or not these added steps will be required is language dependent, however, and, for example, may be more of an issue in C or Fortran than in more object-based languages like Java. But, in any case, these additional memory management steps, if required, should not pose a significant programming obstacle.

The presence of keywords that use this continuation convention files should have no harmful affect on existing FITS software as long as the behavior of that software does not depend on the value of those keywords. Since the continuation keyword records do not formally have a value field, they will be effectively invisible to FITS software that has not been modified to support the continuation convention. Legacy software should simply interpret these header records in the same way as `COMMENT` or `HISTORY` keywords.

We estimate that the cost of retrofitting software to fully support the keyword continuation convention should be relatively small. The biggest cost will be in upgrading the standard FITS libraries that most applications software packages rely on to read and write FITS files so that the library functions can read, write, and delete (and possibly modify in place) keywords that have string values that are continued over multiple header records. The amount of programming effort that will be required to do this depends in part on what computer language is used, but as a benchmark reference, both the CFITSIO library and the Java FITS library were modified to support the HEASARC continuation convention with only a few days or less of programming effort.

After a FITS library is upgraded to support the continuation convention, applications programs that are linked to that library will in many cases inherit the ability to read and write long keyword string values without any further changes. In other cases, however, small modifications to the application program may be needed, for example, to allocate or deallocate the additional memory needed to store the long string values.   Overall, we anticipate that the effort required to support the continuation convention will be similar to, or perhaps slightly larger than, the effort needed to support longer keywords names as was discussed in Section 1.

2.5 Summary

The current 68-character limit on the length of FITS string-valued keywords is significantly impacting the FITS community. Although most FITS projects have been able to successfully work around this restriction (e.g., by developing a local FITS convention for long string values or by simply using `COMMENT` and `HISTORY` keywords), the development of a standard continuation convention, that is recognized and supported by the whole FITS community, would have major benefits. This would allow projects to store character-string meta data in FITS keywords in a clearer and more manageable way, and would provide a standard method for sharing that meta data with other projects. Given the relatively low cost of modifying FITS software to support longer string keyword values, this would appear to be a very worthwhile investment for future astronomical research.

3 The restriction on characters allowed in FITS keyword names

If the definition of the FITS format is modified to allow longer keyword names (as discussed in Section 1), this will open up new possibilities for developing more complex keyword naming conventions. These naming conventions could in turn benefit from having a richer set of allowed characters in the names, in addition to the currently allowed set of 38 characters: uppercase letters A-Z, digits 0-9, the underscore, and the hyphen. More input from the FITS community is needed in order to make an informed decision about which new characters, if any, should be allowed, so we only make a few preliminary observations here.

First, we recommend that any new allowed characters in keyword names should continue to be restricted to the set of 7-bit ASCII text characters (from decimal 32 through 126) in order to maximize backward compatibility with existing FITS files and software. The possibility of supporting U~~u~~nicode ~~characters sets~~ (which can be used to support a wide variety of alphabets and scripts, including but not limited to support Latin extension character sets in ~~uses the~~ full range of 8-bit ASCII characters) is a more complicated topic and should be considered separately.

This leaves 57 other ASCII text characters that could potentially be allowed in keyword names. We have divided these into 3 groups: 1) those that should continue to be prohibited in keyword names; 2) those that would likely be especially useful if allowed in keyword names; and 3) characters that we have no recommendation, as listed below.

1) Some ASCII text characters should continue to be prohibited because they would confuse or complicate the parsing of FITS header record. We recommend excluding the following 15 characters on these grounds

32 (space)

34 " (double quote)

39 ' (single quote)

42 *

44 , (comma)

47 /

58 :

59 ;

60 <

61 =

62 >

63 ?

92 \

96 `

124 |

2) The following 29 characters are most likely to be useful to FITS users:

36 $ (useful as a hierarchy or structure delimiter)

46 . (period; userful as a hierarchy or structure delimiter~~)~~

64 @ (useful as a hierarchy delimiter)

97 - 122 (the set of lowercase letters) Allowed, however keyword names shall be treated as case-in~~IN~~sensitive, so, for example, the names DATAMEAN, DataMean, and datamean shall all be considered equivalent.~~.~~

3) This leaves 13 more ASCII text characters for which we have no strong opinion:~~tions~~:

33 !

35 #

37 %

38 &

40 (

41)

43 + (commonly used in astronomical object names)

91 [

93]

94 ^

123 {

125 }

126 ~

Again, we emphasize that these are only preliminary recommendations, and that the final set of allowed characters should only be decided after much broader input from the FITS community.

3.1 *Impact and costs of expanding the set of allowed characters*

The cost of modifying software to support additional characters in FITS keyword names will likely be relatively low and similar to the costs associated with allowing longer keyword names, as discussed in Section 1. It should be noted, however, that some legacy software applications may be unable to read input FITS files than have keywords that use the expanded character set for keyword name, if that software strictly enforces the current restrictions on the keyword names. Some FITS libraries (including CFITSIO) check that the keyword name conforms to the rules in the FITS Standard before writing or copying it into a FITS file header. As a result, some legacy software applications may not be able to operate on FITS files that contain what it considers to be 'illegal' characters in a keyword name, until the software is upgraded to support the new characters.